IMAC-Sim：存内模拟计算架构的电路级模拟器

Md Hasibul Amin
ma77@email.sc.edu 南卡罗来纳大学 南卡罗来纳州哥伦比亚，美国

Mohammed E. Elbtity
elbtity@ieee.org 南卡罗来纳大学 南卡罗来纳州哥伦比亚，美国

Ramtin Zand
ramtin@cse.sc.edu 南卡罗来纳大学 南卡罗来纳州哥伦比亚，美国

arXiv:2304.09252v1 [cs.ET] 18 Apr 2023

摘要

随着基于忆阻器的存内模拟计算（IMAC）架构日益受到关注，并被视为机器学习应用中高能耗计算系统的替代方案，能够探索其器件级和电路级设计空间的工具将显著推动该领域的研究与开发。因此，本文开发了 IMAC-Sim，这是一款用于 IMAC 架构设计空间探索的电路级模拟器。IMAC-Sim 是一个基于 Python 的仿真框架，可根据用户选择的各种器件级和电路级超参数生成 IMAC 电路的 SPICE网表，并使用用户指定的数据集自动评估所设计电路的准确率、功耗和延迟。此外，IMAC-Sim 还模拟 IMAC 架构中的互连寄生电阻和电容，并配备了水平和垂直分区技术，以克服这些可靠性挑战。IMAC-Sim 是一款灵活的工具，支持广泛的器件级和电路级超参数。本文通过受控实验展示 IMAC-Sim 的若干重要功能，而研究人员可通过开源工具[1]使用其全部功能。

1 引言

机器学习（ML）工作负载的计算需求正在迅速增长，传统的冯·诺依曼架构难以跟上这一增长 [1]，因为处理器与内存之间的数据传输成本高昂。近年来，人们对将存内计算（IMC）作为一种替代方案产生了浓厚兴趣，这种方案能够直接在数据所在的位置执行计算 [2]。新兴的忆阻器器件已广泛应用于存内计算架构，以在机器学习模型中实现矩阵-向量乘法（MVM）运算。虽然忆阻器技术已应用于数字 [3] 和模拟 [4, 5] 存内计算电路，但本文聚焦于存内模拟计算（IMAC）架构，因为它们在实现卓越能效方面具有巨大潜力。例如，Imec 最近提出了一项蓝图，旨在实现每瓦 10,000 万亿次运算（TOPS/W）的机器学习推理 [6]，，该蓝图基于忆阻器 IMAC 架构。

随着人们对 IMAC 架构的关注度不断提升，需要使用仿真框架来探索设计选项并调整超参数，以实现特定的设计目标。现有的存内计算仿真框架可分为两类：架构级 [9, 10]和电路级 [7, 8] 框架。架构级仿真框架使用行为级模型和解析计算来获取各种存内计算架构的特性，而电路级仿真器则利用完整电路分析技术来分析存内计算电路的功能和性能。尽管架构级工具比电路级框架更快，但它们通常会忽略一些重要的设计细节，从而导致仿真结果与实际实现之间出现显著差异。另一方面，完整电路仿真能够以增加仿真时间为代价提供准确的结果。

两个知名的架构级仿真框架是 MNSim [9] 和 NeuroSim [10]，，它们使用解析模型来估算存内计算架构的面积、功耗和延迟。MNSim 验证结果表明，与针对两层全连接神经网络的 SPICE 电路仿真相比，其功耗、能耗和延迟计算的误差约为 5%。然而，MNSim 不报告实际的准确率结果；NeuroSim [10] 则与机器学习仿真器集成，以获取学习准确率和分类准确率数值，但它仍缺少针对存内计算阵列模拟行为的精确电路级预测模型。

DPE [7] 和 CCCS [8] 是用于存内计算电路的两个主要电路级仿真平台。这两个框架都使用在 MATLAB 中开发的电路级求解器。DPE [7] 主要致力于创建一种优化方案，在考虑非理想效应的情况下，将预训练权重映射到忆阻器交叉阵列中。DPE 并不根据阵列尺寸、电阻率、互连厚度等参数计算互连寄生参数，而是将一个固定的预定义数值作为寄生效应仿真的输入。此外，DPE 获取构建模块的功耗和延迟，然后使用解析计算估算整个架构的功耗和延迟，因此预计会出现与架构级仿真器类似的仿真精度损失。另一方面，CCCS [8] 框架采用模型简化技术，通过将交叉阵列划分为若干小型子模块来评估寄生效应对交叉阵列输出的影响，并忽略这些子模块中的寄生效应。尽管创建子模块减少了需要同时处理的节点数量并缩短了仿真时间，但由于对互连压降的估计不准确，会导致计算误差增加。CCCS 验证结果 [8] 表明，这些简化方法能够实现近乎 30×的仿真加速，但代价是最高达 27% 的仿真误差，这可能会严重影响 IMAC 电路的设计与优化。此外，CCCS [8] 框架仅提供机器学习模型的准确率结果，不支持功耗和延迟计算。

本文提出了 IMAC-Sim 框架，用于对存内模拟计算架构进行准确的 SPICE 级建模。IMAC-Sim 是一种高度模块化的参数化工具，

[1]https://github.com/iCAS-Lab/IMAC-Sim

表1：IMAC-Sim 与其他电路级仿真框架的比较

	可靠性				可扩展性	输出		
	寄生效应	噪声	PV	温度	分区	准确率	功耗	延迟
DPE [7]	✓	✓	✓	✓	×	电路仿真	解析模型	
CCCS [8]	✓	×	×	×	×	电路仿真	×	×
IMAC-Sim	✓	✓	✓	×	✓	**完整电路仿真**		

表 2：IMAC-Sim 超参数。

参数	数值
网络拓扑	$T_N = [L_1, L_2, ..., L_n]$
晶体管工艺节点	FinFET、互补金属氧化物半导体
标称电压	$[VDD,\ VSS]$
神经元电路模型	*Sigmoid*函数，双曲正切函数，修正线性单元，等。
突触技术	$[R_{low}, R_{high}]$
垂直分区	$V_P = [vp_1, vp_2, ...,\ vp_{n-1}]$
水平分区	$H_P = [hp_1, hp_2, ...,\ hp_{n-1}]$
突触位单元尺寸	[宽度, 高度]
互连线	[电阻率、厚度、宽度，.....]
采样时间	$t_{sampling}$

能够通过完整的电路 SPICE 仿真报告准确率、功耗和延迟。表1对 IMAC-Sim 与以往的电路级IMC仿真器进行了比较。如表中所列，除寄生效应外，DPE 还会对忆阻器器件的噪声、工艺-电压（PV）变化和温度效应进行仿真，而 CCCS 不考虑其中任何因素。到目前为止，IMAC-Sim 已具备研究噪声和 PV 效应的能力。IMAC-Sim 的独特功能包括：（1）根据版图设计信息以及电阻率、互连线宽度、厚度等互连线参数，自动计算寄生电阻和电容；（2）实现分区技术，将机器学习工作负载分配到分区后的子阵列中，从而降低互连线寄生效应对大规模网络的影响；（3）通过完整的电路 SPICE 仿真，准确报告功耗和延迟。

2 IMAC-SIM 框架

图 1 展示了 IMAC-Sim 框架的框图。该框架基于网络拓扑以及表 2 中列出的由用户定义的特定器件级和电路级超参数，生成 IMAC电路的 SPICE网表。生成 IMAC电路后，IMAC-Sim 通过运行 SPICE 仿真，在开发的 IMAC 电路上自动执行各种机器学习任务；相应的数据集由用户提供。它提取电路的输出，并将其与目标标签进行比较，以评估 IMAC 的准确率。此外，它还测量功耗和延迟，并将其与计算得到的准确率一同作为输出提供给用户。用户可以利用这些信息优化其开发的电路。IMAC-Sim 框架包含三个主要的 Python模块：*testIMAC*、*mapIMAC* 和 *mapLayer*，如图 1 所示，具体描述如下。

testIMAC 模块作为父文件运行，用于控制深度神经网络（深度神经网络）工作负载在 IMAC 架构上的部署，并评估准确率、功耗和延迟。*testIMAC* 模块接收用户提供的两组输入；一组

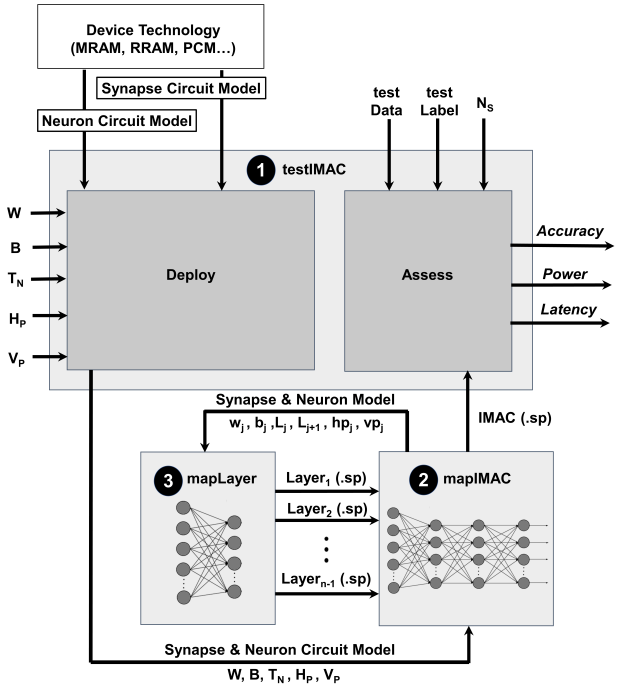


图 1：IMAC-Sim 框架的框图。

用于部署，另一组用于评估。在部署阶段，它接收预训练的 *weight* 和 *bias* 矩阵、网络拓扑（T_N）、每层的水平分区数量（H_P）和垂直分区数量（V_P），以及用于在 IMAC架构上部署深度神经网络的器件级和电路级超参数。在评估阶段，它接收测试数据集（*TestData*）、测试标签（*TestLabel*）以及测试样本数量（N_S），用于评估所开发的 IMAC 电路。 T_N是一个数据阵列，包含深度神经网络从输入层到输出层的节点数量。H_P和 V_P是两个阵列，分别存储每层的水平分区数和垂直分区数。*testIMAC* 模块的功能如算法1所示。

如算法1所示，对于每个输入样本，*testIMAC* 首先将目标标签存储在名为 *label* 的数组中。随后，它会调用另一个 Python 模块 *mapIMAC*，该模块负责创建 IMAC 电路的 SPICE 网表。为此，*mapIMAC* 会调用另一个 Python 模块 *mapLayer*，该模块为深度神经网络的每一层构建独立的子电路，包括层间寄生效应以及用户通过 H_P 和 V_P 数组指定的分区要求。我们将在接下来的两个小节中详细讨论寄生效应和分区。*mapLayer* 模块会将所有层子电路的 SPICE 文件返回给*mapIMAC*，

算法1： 使用 *IMAC-Sim* 实现 IMAC 框架

/* *Module* ❶
输入： 测试数据集 (*TestData*)，测试标签 (*TestLabel*)，权重 (*W*)、偏置 (*B*)、网络拓扑 (T_N)、编号。的测试样本 (N_S)、水平分区 (H_P)、垂直分区 (V_P)，R_{low}, R_{high}

初始化： $Error = 0$, $PW\ R = 0$, $W = [w_1, w_2, ..., w_{n-1}]$, $B = [b_1, b_2, ..., b_{n-1}]$, $T_N = [L_1, L_2, ..., L_n]$, $H_P = [hp_1, hp_2, ..., hp_{n-1}]$, $V_P = [vp_1, vp_2, ..., vp_{n-1}]$
对于 $i = 1$ **到** N_S **执行**
  标签 ⇐ 输入 *i*的目标标签
  /* *Module* ●
  mapIMAC (W, B, T_N, H_P, V_P)**:**
    对于 $j = 1$ **到** $j = len(T_N) - 1$ **执行**
      /* *Module* ●
      mapLayer ($w_j, b_j, L_j, L_{j+1}\ hp_j, vp_j$)**:**
        实现分区（hp_j，vp_j）
        插入互连寄生参数
      返回 $Layer_j$ *SPICE* 子电路
    end
    拼接这些 $layer_j$ 子电路
  返回 *IMAC SPICE* 电路
  运行*IMAC* 电路的 SPICE 仿真
  输出 ⇐ IMAC 电路的输出
  PWR += IMAC 电路的功耗
  如果 (*out*≠标签)**则**
    $Error$ += 1;
  end
结束
次打印 $ErrorRate = Error/N_S$
次打印 $P_{average} = PW\ R/N_S$

后者将这些文件拼接起来，生成主 *IMAC* SPICE 文件。最后，*testIMAC* 使用由测试数据集生成的输入电压，对生成的 *IMAC*SPICE 文件运行 SPICE 仿真，提取 IMAC 电路最后一层的输出（*out*），并将其与 *label* 进行比较，以计算准确率。此外，*testIMAC* 还会测量该电路在各种输入下的平均功耗和延迟，并将结果报告给用户。

2.1 互连寄生参数

互连寄生电阻（R_W）和电容（C_W）取决于 IMAC 子阵列中互连线的导线几何结构和材料特性。扩大 IMAC 阵列的规模会增加 R_W和 C_W，进而可能增加 IMAC 电路的延迟并限制其工作时钟频率。此外，增加的 R_W会降低 IMAC 的读裕量，从而影响其准确率 [11, 12]。IMAC 的互连寄生电阻可通过以下方程求得：

$$R_W = \rho \frac{L}{W.T}, \tag{1}$$

算法 2： 分区 (h_p,v_p)

输入： 权重 (w)、偏置 (b)、层维度 (L_1, L_2)，数量。水平分区 (h_p) 的数量，垂直分区（v_p）

对于 $i = 1$ **到** h_p **执行**
  $NH_i \Leftarrow L_1 + 1$
  $NHP_i \Leftarrow NH_i \times i/h_p + min((NH_i \bmod h_p)/i,\ 1)$
  对于 $j = 1$ **到** v_p **执行**
    $NVP_j \Leftarrow L_2 \times j/v_p + min((L_2 \bmod v_p)/j,\ 1)$
    /* *Partition* P_{ij}
    放置一个尺寸为 $NHP_i \times NVP_j$
  end
end
的二维矩阵

其中，ρ、L、W和 T分别为金属线的电阻率、长度、宽度和厚度。虽然通常认为特定金属的电阻率是固定参数，但对于 10 纳米以下技术节点，金属宽度接近电子平均自由程，电阻率会因表面散射和晶界散射而增大 [13]。如下文所述，这两种众所周知的散射效应可分别通过 Fuchs-Sondheimer（FS）[14] 和 Mayadas-Shatzkes（MS）[15] 模型进行量化，

$$\frac{\rho}{\rho_{Cu}} = \frac{3}{4}(1-p)\frac{l_0}{W} + \left[1 - \frac{3\alpha}{2} + 3\alpha^2 - 3\alpha^3 \ln\left(1 + \frac{1}{\alpha}\right)\right]^{-1} \tag{2}$$

其中，$\alpha = \frac{l_0}{d}\frac{R}{1-R}$、$\rho_{Cu}$是块体铜的电阻率（$1.9\times10^{-9}\,\Omega m$），$l_0$ 是铜中电子的平均自由程（39 nm），W是金属互连的宽度，p 是镜面散射分数，d是平均晶粒尺寸，R是电子在晶界处反射的概率。假设 R和 p分别为 0.3 和 0.25，并根据文献中提到的平均值 [16],[17]，假设 d等于互连线宽度。

寄生电容在决定 IMAC 的延迟方面也发挥着重要作用。这里，我们使用 Sakurai-Tamaru模型 [18]计算单位长度的寄生电容，如下方公式所述：

$$\begin{aligned} C_W = &\frac{\epsilon}{2}\left[1.15\left(\frac{W}{H}\right) + 2.8\left(\frac{W}{H}\right)^{0.222}\right] + 2\epsilon\left(\frac{S}{H}\right)^{-1.34} \\ &\times\left[0.03\left(\frac{W}{H}\right) + 0.83\left(\frac{T}{H}\right) - 0.07\left(\frac{T}{H}\right)^{0.222}\right] \end{aligned} \tag{3}$$

其中，$\epsilon = 20\epsilon_0$ 是金属间空间的介电常数，W和 T分别是金属线的宽度和厚度，$H = 20nm$是金属层间距，S是线间距 [11]。IMAC-Sim 利用公式（2）和（3）计算互连寄生参数，并将其注入 IMAC电路中。

2.2 水平和垂直分区

由于互连寄生参数会严重降低 IMAC 电路的准确率，文献中提出了水平与垂直分区技术

表 3：为限制 IMAC 设计空间而在本文中固定的超参数。

参数	数值
晶体管工艺节点	14纳米鳍式场效应晶体管
标称电压	[VDD = 0.8伏特，VSS= −0.8伏特]
神经元电路模型	Sigmoid函数
互连线	电阻率 $\rho = 1.9 \times 10^{9}\Omega.m$ 厚度= $22nm$ 宽度= $4\lambda = 36nm$
金属层间距	$20nm$

以降低 IMAC 电路中的 [19, 11] R_W和C_W。我们为 IMAC-Sim 配备了水平与垂直分区功能，并在算法 2 中进行了说明。用户可以通过 *testIMAC* 模块中的 H_P和 V_P阵列，分别调整深度神经网络中每一层的水平与垂直分区数量这一超参数。*testIMAC* 模块会分别为每一层调用 $partitioning(h_P, v_P)$，自动对该层进行分区并连接外围电路。如算法 2 所示，分区函数遍历 h_p和v_P，并依次连接所有分区。 NH_i表示水平方向上的位单元数量。 NHP_i和 NVP_j分别表示水平方向和垂直方向上每个分区中的位单元数量。因此，会为分区 P_{ij}连接一个维度为 $NHP_i \times NVP_j$的位单元二维矩阵。

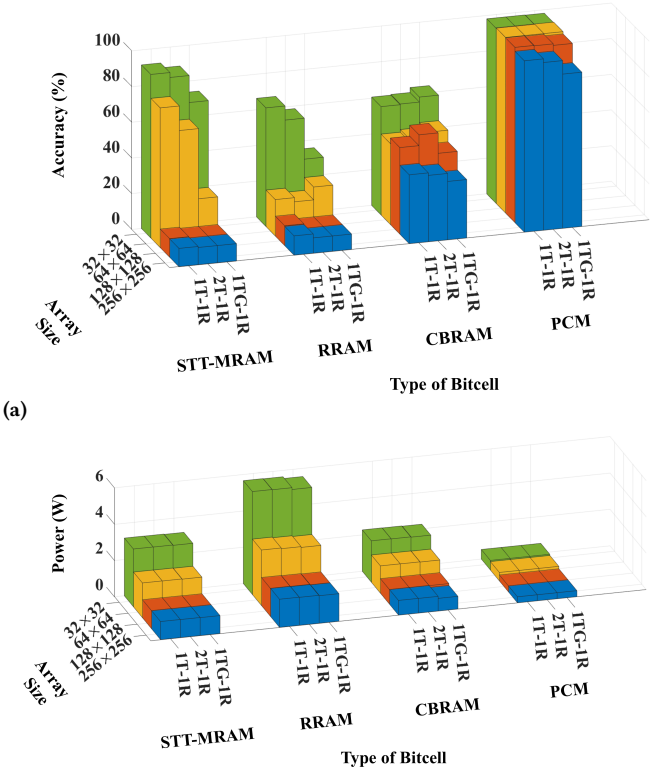
3 仿真结果与讨论

在本节中，我们利用所开发的 IMAC-Sim 框架，使用包含 20×20 像素手写数字图像的 MNIST 数据集，实现了一个用于分类应用的 $400 \times 120 \times 84 \times 10$ 深度神经网络模型。需要强调的是，我们在此并不关注设计一种经过优化的 IMAC架构，而是展示 IMAC-Sim 工具的一些重要特性和功能。因此，针对不同类型的数据集在 IMAC 架构上开发各种深度神经网络模型不在本文讨论范围之内。尽管如此，我们仍欢迎研究人员利用项目 GitHub 仓库中提供的开源工具，对其 IMAC 设计进行基准测试，并指导早期阶段的设计决策。

3.1 限制实验的设计空间

在此，为了限制广泛的 IMAC 设计空间，我们固定了表 3 中列出的一些超参数，同时在下一小节中研究其他参数对 IMAC 准确率和功耗的影响。我们使用 14nm 高性能 PTM-MG FinFET 模型，以及电压分别为 0.8V 和 -0.8V 的 VDD和 VSS。基于 PTM 14nm FinFET 模型的 18nm 栅长和 22nm Fin 高度，[20]，版图设计参数 λ 和金属厚度分别固定为 9nm 和 22nm。

我们采用一种模拟 sigmoid 神经元，其中包含两个电阻器件和一个基于 CMOS 的反相器。神经元电路中的电阻器件构成分压器，降低了反相器线性工作区的斜率，从而使输出电压平滑地从高电平转换为低电平，进而实现



(b) 图 2：在具有不同子阵列尺寸、忆阻器技术和位单元类型的 IMAC 上部署 $400 \times 120 \times 84 \times 10$ 深度神经网络模型所得到的结果。(a) 准确率和 (b) 功耗。

sigmoid 激活函数 [21]。用户可以创建名为“neuron.sp”的 SPICE 子电路文件，并将其放在 IMAC-Sim 工具所在的同一目录中，以使用其他任何神经元电路。通过固定互连参数以及突触和神经元结构，我们将 IMAC 广阔的设计空间限定在一定范围内，以便重点介绍下文所述的 IMAC-Sim 的一些有趣特性。

3.2 设计空间探索

在本节中，我们使用 IMAC-Sim，针对 $400 \times 120 \times 84 \times 10$ 深度神经网络模型，探索不同器件技术、突触位单元尺寸和分区数量下 IMAC 的设计空间。突触位单元尺寸是估算互连线之间距离并确定金属线长度的重要超参数，这会影响互连寄生电阻和电容，如第 2.1 节所述。我们针对三种不同类型的位单元探索结果：1) 1T-1R（长度： 15λ，宽度： 12λ）、2) 2T-1R（长度： 15λ，宽度： 12λ） [22] ,以及 3) 1TG-1R（长度： 15λ，宽度： 12λ） [23]。此外，我们还针对四种不同的器件技术开展实验：1) 磁阻随机存取存储器（MRAM）（R_{low}： $8.5k\Omega$， R_{high}： $25.5k\Omega$） [24], 2; 2) 阻变随机存取存储器（RRAM）（R_{low}： $2.5k\Omega$， R_{high}： $100k\Omega$）[25], 3; 3) 导电桥接随机存取存储器（CBRAM）（R_{low}： $10k\Omega$， R_{high}： $1M\Omega$） [26], ; 以及 4) 相变存储器（PCM）（R_{low}： $78k\Omega$， R_{high}： $202k\Omega$） [27]。

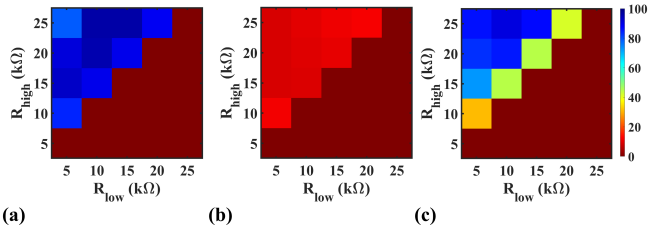


图 3：IMAC-Sim 获得的准确率结果：(a) 无寄生效应，(b) 有寄生效应，(c) 有寄生效应且采用分区。

此外，我们还通过改变水平分区和垂直分区的数量，研究分区的影响。虽然 IMAC-Sim 支持任意的水平和垂直分区值，但在这里，我们根据不同尺寸 IMAC 子阵列的最大利用率，为每一层选择分区数量。例如，如果使用 256 × 256 子阵列，则包含 400 个输入的第一层必须划分为两个水平分区，才能装入两个 256 × 256 IMAC 子阵列，而其他层无需进行水平和垂直分区即可装入阵列。我们针对不同的器件类型和位单元尺寸，在每种分区场景下运行 IMAC-Sim，以测量准确率和功耗，结果如图 2 所示。图 2 所示结果表明，在尺寸更小的 IMAC 子阵列中，随着水平分区和垂直分区数量的增加，准确率和功耗也会随之增加。例如，在 32 × 32 IMAC 子阵列上高度分区部署深度神经网络模型时，基于 MRAM 的 1T-1R 位单元可达到 91.6% 的准确率，而在 256 × 256 相同类型的子阵列上采用较少分区时，准确率仅为 11%。然而，在 32×32 IMAC 子阵列中，较少分区时的功耗为 0.996W，而较多分区时功耗最高可达 3.18W。此外，如果考虑 1TG-1R 位单元，准确率会从上述的 91.6% 降至 73.72%，这表明位单元尺寸越大，IMAC 受寄生效应的影响越大。此外，对于采用类似布局的基于 PCM 的位单元，准确率提高至 98.56%，功耗为 0.52W。

为了研究器件技术对 IMAC架构性能的影响，我们扫描 R_{low} 和 R_{high}的数值，并使用 IMAC-Sim 在三种不同场景下获取深度神经网络准确率，如图 3 所示。图中右下角的准确率数值始终为零，因为当 R_{high} 小于或等于 R_{low}时，电路会发生故障。图 3a 中的模型准确率较高，这是因为我们忽略了寄生电阻的影响。随后，我们将寄生效应加入电路，这显著影响了 DNN 精度，如图 3b 所示。图 3c 展示了将深度神经网络模型分区部署到 32×32 个 IMAC子阵列上的结果。显然，分区提升了因寄生效应而严重下降的准确率。例如，当 $R_{low} = 5k\Omega$ 和 $R_{high} = 15k\Omega$ 时，不考虑寄生效应的准确率为 93.3%；考虑寄生效应后，准确率降至 8.9%，而分区使其提升至 72.7%。图 3 表明，分区带来的准确率提升程度很大程度上取决于 R_{low} 和 R_{high} 的数值。因此，IMAC-Sim 可帮助设计人员在早期阶段就器件技术的选择做出决策。

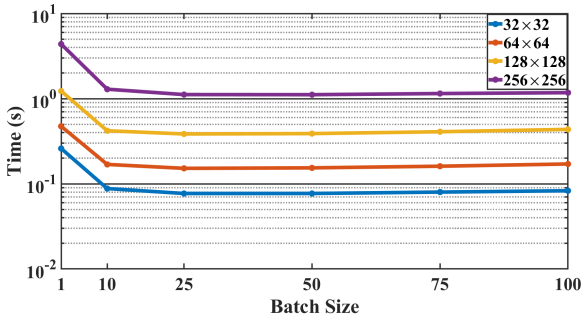


图 4：不同阵列大小下，批量大小与平均仿真时间的关系。

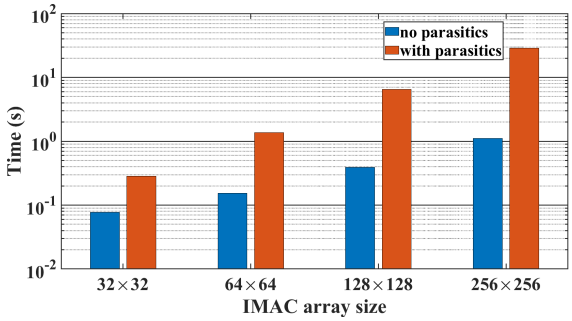


图 5：批量大小为 50 时，不同阵列大小下有无寄生效应的平均仿真时间。

3.3 仿真时间分析

本节中，我们在 Intel Xeon Silver 4210R CPU 上针对各种 IMAC子阵列运行 IMAC-Sim 框架，并收集仿真时间信息。为缩短仿真时间，我们让 IMAC-Sim 能够在每次 SPICE 仿真中运行多项测试，方法是将多项测试的输入电压拼接成一个分段线性（PWL）函数。在此，我们定义一个“批量大小”参数，表示每次仿真所拼接的输入测试数量，例如 MNIST 图像。为了在 SPICE 中实现批处理，我们使用 PWL信号，根据测试用例每隔 1ns 改变一次电压等级。例如，批量大小为 50 时，输入波形长度为 50ns。1ns 同时也是记录输出的采样间隔。

在图 4 中，我们展示了运行不同尺寸的 IMAC 子阵列时，在不考虑互连寄生参数的情况下，不同批量大小对应的平均仿真时间。结果表明，仿真时间会在特定批量大小下达到最小值。例如，批量大小为 1 时，仿真一个 256×256 IMAC 子阵列所需的平均时间为 4.37 秒。批量大小为 50 时，平均仿真时间降至 1.11 秒；批量大小为 100 时，则增至 1.18 秒。图 5 展示了批量大小为 50 时，不同阵列尺寸在考虑和不考虑寄生效应情况下的仿真时间。结果表明，考虑互连寄生参数进行电路仿真，可能会使仿真时间增加约一个数量级。根据图 5 所示结果，考虑寄生效应时，256×256 IMAC 子阵列的平均仿真时间为 28.76 秒，而不考虑寄生效应时仅为 1.11 秒。

4 结论

本文介绍了 IMAC-Sim，这是一种基于 Python 的仿真框架，用于实现 IMAC 架构的 SPICE 电路。IMAC-Sim 是一款灵活的工具，支持多种器件级和电路级超参数，可用于设计基于忆阻器的 IMAC 架构。与以往面向存内计算架构的电路仿真器相比，IMAC-Sim 的一些突出特点包括：向用户提供深度神经网络准确率报告；涵盖互连寄生参数等可靠性挑战；并提供分区等解决方案，以克服这些挑战。鉴于 IMAC-Sim 支持广阔的 IMAC 设计空间，我们通过固定部分超参数来约束设计空间，并开展实验以展示 IMAC-Sim 的一些独特优势。实验考察了分区以及各种忆阻器件技术对 IMAC 架构准确率和功耗的影响。IMAC-Sim 得出的结果揭示了一些重要权衡，可在设计早期阶段提供给开发者，展现了 IMAC-Sim 对电路设计师和计算机体系结构师的潜在价值。IMAC-Sim 的开发为未来工作带来了多种可能性，包括将 IMAC-Sim 与多目标优化算法集成，以针对各种机器学习工作负载实现优化的 IMAC 设计。

参考文献

[1] Yu Wang、Gu-Yeon Wei 和 D. Brooks。2019。深度学习的 TPU、GPU 和 CPU 平台基准测试。*ArXiv*，abs/1907.10701。[2] A. Sebastian 等。2020。用于存内计算的存储器件及其应用。*Nature Nanotechnology*，15，529–544。[3] S. Jain 等。2018。采用自旋转移矩磁性随机存取存储器的存内计算。*IEEE Transactions on Very Large Scale Integration (VLSI) Systems*，26，3，470–483，doi：10.1109/TVLSI.2017.2776954。[4] Mohammed Elbtity 等。2021。用于移动设备上高能效 CNN 推理的存内模拟计算协处理器。*2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2021*。[5] Aayush Ankit 等。2019。Puma：用于机器学习推理的可编程超高效忆阻器加速器。载于 *Proceedings of the T wenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*（ASPLOS '19）。Association for Computing Machinery，美国罗德岛州普罗维登斯，715–731。[6] S. Cosemans 等。2019。面向 10000tops/w 深度神经网络推理的模拟存内计算——电路蓝图、器件选项及要求。载于 *2019 IEEE International Electron Devices Meeting (IEDM)*，22.2.1–22.2.4。doi：10.1109/IEDM19573.2019.8993599。[7] Miao Hu 等。2016。神经形态计算的点积引擎：通过编程 1*2016 53*1 1 6 10 1145 t m 交叉阵列加速矩阵-向量乘法。载于 *nd ACM/EDAC/IEEE DesignAutomation Conference (DAC)*，–。doi：10.1145/2897937.2898010。[8] Fan Zhang 和 Miao Hu。2020。CCCS：面向存内计算的定制 SPICE 级交叉阵列电路仿真器。载于 *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*，1–8。[9] Lixue Xia 等。2016。MNSIM：基于忆阻器的神经形态计算系统仿真平台。载于 *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*，469–474。[10] Pai-Yu Chen、Xiaochen Peng 和 Shimeng Yu。2018。NeuroSim：用于在线学习神经启发式架构基准测试的电路级宏模型。*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*，37，12，3067–3080。doi：10.1109/TCAD.2018.2789723。[11] F. L. Aguirre 等。2020。准静态忆二极管模型在交叉点阵列中的应用：面向大型数据集的模式识别。*IEEE Access*，8，202174–202193。doi：10.1109/ACCESS.2020.3035638。[12] Md Hasibul Amin、Mohammed Elbtity 和 Ramtin Zand。2022。全模拟存内计算架构中的互连寄生参数与分区。载于 *2022IEEE InternationalSymposium onCircuitsand Systems(ISCAS)*，389–393。doi：10.1109/ISCAS48785.2022.9937884。[13] Daniel Josell、Sywert H. Brongersma 和 Zsolt Tőkei。2009。纳米尺度互连中的尺寸相关电阻率。*Annual Review of Materials Research*，39，1，231–254。doi：10.1146/annurev-matsci-082908-145415。[14] K. Fuchs。1938。根据金属电子理论研究薄金属膜的电导率。*Mathematical Proceedings of the Cambridge Philosophical Society*，34，1，100–108。doi：10.1017/S0305004100019952。[15] A. F. Mayadas 和 M. Shatzkes。1970。多晶膜的电阻率模型：外部表面任意反射的情况。*Phys. Rev. B*，1，1382–1389。doi：10.1103/PhysRevB.1.1382。https://link.aps.org/doi/10.1103/PhysRevB.1.1382。[16] S. Rossnagel 和 Tung-Sheng Kuan。2004。尺寸效应区域中铜电导率的变化。*Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*，22，（2004 年 1 月）。doi：10.1116/1.1642639。[17] W. Steinhögl 等。2004。对横向尺寸为 100 nm 及以下铜导线电阻率的综合研究。*Journal of Applied Physics*，97，（2004 年 12 月），023706–023706。doi：10.1063/1.1834982。[18] T. Sakurai 和 K. Tamaru。1983。二维和三维电容的简明公式。*IEEE Transactions on Electron Devices*，30，2，183–185。doi：10.1109/T-ED.1983.21093。[19] Md Hasibul Amin、Mohammed E. Elbtity 和 Ramtin Zand。2022。Xbar分区

模拟存内计算电路中应对寄生效应和提高噪声容限的一种实用方法。新兴与重点电路及系统*IEEE*期刊，12，4，867–877。doi：10.1109/JETCAS.2022.3222966。[20] Saurabh Sinha 等。2012。利用预测性工艺模型探索 20 纳米以下 FinFET设计。载于*DAC*设计自动化会议 *2012*，283–288。doi：10.1145/2228360.2228414。[21] Md Hasibul Amin 等。2022。用于存内计算的基于MRAM的模拟Sigmoid函数。载于*2022*年大湖区*VLSI*研讨会论文集（GLSVLSI '22）。美国计算机协会，美国加利福尼亚州尔湾，319–323。ISBN：9781450393225。doi：10.1145/3526241.3530376。https://doi.org/10.1145/3526241.3530376。[22] Sumeet Kumar Gupta 等。2012。自旋转移矩MRAM的版图感知优化。载于*2012*年欧洲设计、自动化与测试会议及展览（*DATE*），1455–1458。doi：10.1109/DATE.2012.6176595。[23] R. Zand、A. Roohi 和 R. F. DeMara。2017。用于自旋霍尔效应MRAM器件的节能且具备工艺偏差鲁棒性的写入电路方案。*IEEE*超大规模集成（*VLSI*）系统学报，25，9，2394–2401。ISSN：1557-9999。doi：10.1109/TVLSI.2017.2699579。[24] Ramtin Zand、Arman Roohi 和 Ronald F DeMara。2018。磁隧道结的基础、建模与应用。纳米尺度器件：物理、建模及其应用，337。[25] Can Li 等。2018。利用大型忆阻器交叉阵列进行模拟信号与图像处理。自然电子学，1，1，52。[26] Yuhan Shi 等。2018。利用亚量子CBRAM阵列进行神经启发式无监督学习与剪枝。自然通讯，9，1，1–11。[27] Benedikt Kersting 等。2020。投影式相变存储器中的状态依赖性与电阻的时间演化。科学报告，10，（2020年5月），8248。doi：10.1038/s41598-020-64826-3。